UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/816,144	03/31/2004	Ulrich Bortfeld	42P19079	9012

8791 7590 03/29/2006

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
IWASHKO, LEV	

ART UNIT	PAPER NUMBER
2186	

DATE MAILED: 03/29/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/816,144	BORTFELD, ULRICH	
	Examiner	Art Unit	
	Lev I. Iwashko	2186	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *31 March 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-25* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-25* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *31 March 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 102

1. The following are quotations of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-6, 9-11, and 14-22 are rejected under U.S.C. 102(b) as being anticipated by Quattromani et al. (US Patent 5,835,949).

Claim 1. An instruction memory comprising:

- memory locations, directly instruction-word addressable, to store linear sequences of instruction data in logically contiguous memory locations of increasing logical word address, including instructions that are one or multiple instruction words in size, wherein instructions stored in the memory locations may be fetched in a single instruction cycle; *(Column 8, lines 39-53 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222. Multiplexers 220 and 222 have inputs coupled to either the instruction cache 65 (a.k.a. L0), the unified cache 60 (a.k.a. L1), or to external memory through the bus interface unit (BIU) 90. Since the instruction cache 65 is organized thirty two bytes (256 bits) wide, multiplexer 224a-224b reduces the cache line down to sixty-four-bit chunks for multiplexers 220 and 222, one-hundred-twenty-eight bits at a time. Similarly, multiplexer 226a-226b reduces the cache line down from the unified cache 60 to sixty-four-bit chunks for multiplexers 220 and 222. The third set of inputs to multiplexers 220 and 222 originate from the BIU 90 which supplies instruction data*

from external memory". Column 4, lines 39-41 – State the following: "It is dual ported (through banking) to permit two memory accesses (data read, instruction fetch, or data write) per clock")

- memory control logic to indicate a current valid read word address of the memory locations; *(Column 2, lines 55-59 – State the following: "Another feature of the present invention is the use of separate Valid and Read Valid bits to accommodate non-cacheable instruction data, and external and higher order cache replacement snoops without flushing the execution pipeline(s)")*
- and a permutation unit to receive instruction data read from the memory locations and order the instruction data according to an order of sequential operation. *(Although the art does not specifically reference a "permutation unit", the multiplexing unit described above does in fact perform the same function as the applicant's permutation unit.)*

Claim 2. An instruction memory according to claim 1, wherein the memory locations are byte addressable. *(Column 10, lines 1-2 – State the following: "index tag Hit I and index tag Hit II signals could be eight bits wide)*

Claim 3. An instruction memory according to claim 1, wherein the memory locations are two-byte addressable. *(Column 7, lines 21-23 – State the following: "A VL/ISA interface chip 91 (such as an HT321) provides standard interfaces to a thirty-two bit VL bus and a sixteen bit ISA bus")*

Claim 4. An instruction memory according to claim 1, wherein the memory locations further comprise multiple banks of addressable locations, each bank having multiple directly addressable lines of instruction data. *(Column 11, lines 38-43 – State the following: "During clock cycle one, program execution continues with prefetch and prefetch+8 linear addresses 7e010 and 7e018 respectively, resulting in hits in the instruction cache 65 on line five, quadrants C and D respectively. Accordingly, Hit I*

and Hit II index tags "tag" the instruction data located at these addresses with the instruction cache line number")

Claim 5. An instruction memory according to claim 4, wherein each bank is one instruction word in width. *(Column 8, lines 39-41 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222")*

Claim 6. An instruction memory according to claim 1, wherein the memory control logic further comprises a pair of memory location pointers to define a window of valid instruction data. *(Column 5, lines 26-28 – State the following: "In ID1, the length of two instructions is decoded (one each for the X and Y execution pipelines) to obtain the X and Y instruction pointers")*

Claim 9. A memory structure for storing variable-instruction-size instructions comprising:

- multiple rows of word-addressable instruction data memory locations, *(Column 8, lines 24-31 – State the following: "The flow of instructions through the pipeline is controlled by the pipe control unit 28. In the preferred embodiment, a single pipe control unit 28 is used to control the flow of instructions through both (or all) of the pipes. To control the flow of instructions through the pipes, the pipe control unit 28 receives "delay" signals from the various units comprising the X and Y pipes, and issues "stall" signals to the various units." The above shows that there are pipes (or rows))*
- the rows of memory locations to receive instruction data in lexical order *(Column 7, lines 50-55 – State the following: "During internal clock cycle 132, instructions X3 and Y3 are in the ID1 stage, instructions X2 and Y2 are in the ID2 stage, instructions X1 and Y1 are in the AC1 stage and instructions X0 and Y0 are in the AC2 stage.*

The instructions continue to flow sequentially through the stages of the X and Y pipelines")

- from a system level-1 (L1) cache *(Column 8, lines 43-44 – State the following: "The unified cache 60 (a.k.a. L1)")*
- and store the instruction data in lexical order in the rows of memory locations, *(Column 8, lines 31-35 – State the following: "If the instruction data originates from the unified cache 60 (L1) or from main memory via BIU 90, it is also concurrently stored in the instruction cache 65 and assigned an index tag corresponding to its instruction cache line number")*
- wherein the instruction data is retrievable in a single instruction clock cycle; *(Column 7, line 64 – States the following: "As shown, no stage requires more than one clock cycle")*
- and control logic coupled with the rows of memory locations, to provide control signals to cause reading and writing of memory locations, and to order the instruction data read from the memory locations in lexical execution order. *(Column 8, lines 24-31 – State the following: "The flow of instructions through the pipeline is controlled by the pipe control unit 28. In the preferred embodiment, a single pipe control unit 28 is used to control the flow of instructions through both (or all) of the pipes. To control the flow of instructions through the pipes, the pipe control unit 28 receives "delay" signals from the various units comprising the X and Y pipes, and issues "stall" signals to the various units." Column 7, lines 14-17 – State the following: "Controller 82 interfaces directly to the thirty-two bit address bus PADDR, and includes a one bit wide data port (not shown) for reading and writing registers within the controller")*

Claim 10. A memory structure according to claim 9, wherein the rows of memory location to store the instruction data in lexical order further comprises storing logically separable units of instruction data in sequential memory

locations spanning multiple rows. *(Column 2, lines 55-59 – State the following: "Another feature of the present invention is the use of separate Valid and Read Valid bits to accommodate non-cacheable instruction data, and external and higher order cache replacement snoops without flushing the execution pipeline(s)")*

Claim 11. A memory structure according to claim 9, wherein the control logic further comprises address indicators to indicate at least a top and a bottom of a window of valid instruction data. *Column 11, lines 1-23 – State the following: "In this example, program flow occurs top to bottom with arbitrary change of flow (COF) instructions occurring in address space 7e00f-7e010 and in address space 7e040-7e047. To simplify explanation, the address space along the cache line in instruction cache 65 is broken down into eight-byte wide "quadrants" designated A, B, C, and D." Column 8, lines 39-41 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222")*

Claim 14. An electronic system comprising:

- a processor having a sequencer *(Column 4, lines 50-58 – State the following: "The CPU core 20 is a super-scalar design with two execution pipelines X and Y. It includes an instruction decoder 21, address calculation units 22X and 22Y, execution units 23X and 23Y, and a register file 24 with thirty-two, thirty-two bit registers. An AC control unit 25 includes a register translation unit 25a with a register scoreboard and register renaming hardware. A microcontrol unit 26, including a microsequencer and microROM, provides execution control")*
- coupled to a level-zero cache to store variable-instruction-size instructions, *(Column 8, lines 43 – States the following: "instruction cache 65 (a.k.a. L0)")*

- the level zero (L0) cache including instruction-word addressable memory locations to store a series of instruction data in execution order, each instruction of the series of instruction data retrievable in a single instruction clock cycle, the L0 cache having read and write logic to write the instruction data to the memory locations, read the data from the written memory locations, and align the data in execution order; *(Column 8, lines 39-53 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222. Multiplexers 220 and 222 have inputs coupled to either the instruction cache 65 (a.k.a. L0), the unified cache 60 (a.k.a. L1), or to external memory through the bus interface unit (BIU) 90. Since the instruction cache 65 is organized thirty two bytes (256 bits) wide, multiplexer 224a-224b reduces the cache line down to sixty-four-bit chunks for multiplexers 220 and 222, one-hundred-twenty-eight bits at a time. Similarly, multiplexer 226a-226b reduces the cache line down from the unified cache 60 to sixty-four-bit chunks for multiplexers 220 and 222. The third set of inputs to multiplexers 220 and 222 originate from the BIU 90 which supplies instruction data from external memory")*
- and a power storage cell coupled with the processor to provide power to the processor. *(Column 2, lines 7-21 – State the following: "Heretofore, techniques which have addressed self-modifying code in pipelined/cached processor designs have employed a content addressable memory (CAM) to store and compare the linear and physical address tags of each instruction queued in the execution pipeline or stored in the instruction cache. Each cell in the CAM therefore, requires a lengthy comparator, a lengthy linear address tag, and a lengthy physical address tag, requiring more die area and taking up valuable space on an designer to make tradeoffs between the*

depth of the execution pipeline and the number of CAMs. Moreover, the CAMs in the execution pipeline are some what duplicative with CAMs already in the instruction cache, consuming more power and compounding power dissipation problems within the integrated circuit.)

Claim 15. A system according to claim 14, wherein the memory locations are byte addressable. *(Column 10, lines 1-2 – State the following: "index tag Hit I and index tag Hit II signals could be eight bits wide)*

Claim 16. A system according to claim 14, wherein the memory locations are two-byte addressable. *(Column 7, lines 21-23 – State the following: "A VL/ISA interface chip 91 (such as an HT321) provides standard interfaces to a thirty-two bit VL bus and a sixteen bit ISA bus")*

Claim 17. A system according to claim 14, wherein the memory locations further comprise multiple banks of addressable locations, each bank having multiple directly addressable lines of instruction data. *(Column 11, lines 38-43 – State the following: "During clock cycle one, program execution continues with prefetch and prefetch+8 linear addresses 7e010 and 7e018 respectively, resulting in hits in the instruction cache 65 on line five, quadrants C and D respectively. Accordingly, Hit I and Hit II index tags "tag" the instruction data located at these addresses with the instruction cache line number")*

Claim 18. A system according to claim 17, wherein each bank is one instruction word in width. *(Column 8, lines 39-41 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222")*

Claim 19. A system according to claim 14, wherein the read logic further comprises memory location pointers to define a window of valid instruction data. *(Column 5, lines 26-30 – State the following: "In ID1, the length of two instructions is decoded (one each for the X and Y execution pipelines) to*

obtain the X and Y instruction pointers--a corresponding X and Y bytes-used signal is sent back to the prefetch buffer 30 (which then increments for the next sixteen byte transfer)")

Claim 20. A system according to claim 19, wherein a read request for a memory location outside the window of valid instruction data causes the processor to fetch more instruction data to alter the window of valid instruction data until the requested memory location is within the window. *(Column 2, lines 55-59 – State the following: "Another feature of the present invention is the use of separate Valid and Read Valid bits to accommodate non-cacheable instruction data, and external and higher order cache replacement snoops without flushing the execution pipeline(s)")*

Claim 21. A method comprising:

- storing units of instruction data in logically sequential memory locations of a level-0 (L0) storage structure; *(Column 8, lines 39-53 – State the following: "The prefetch buffer 30 receives a lower quad-word (bits <63:0>) from multiplexer 220 and an upper quad-word (bits <127:64>) from multiplexer 222. Multiplexers 220 and 222 have inputs coupled to either the instruction cache 65 (a.k.a. L0), the unified cache 60 (a.k.a. L1), or to external memory through the bus interface unit (BIU) 90. Since the instruction cache 65 is organized thirty two bytes (256 bits) wide, multiplexer 224a-224b reduces the cache line down to sixty-four-bit chunks for multiplexers 220 and 222, one-hundred-twenty-eight bits at a time. Similarly, multiplexer 226a-226b reduces the cache line down from the unified cache 60 to sixty-four-bit chunks for multiplexers 220 and 222. The third set of inputs to multiplexers 220 and 222 originate from the BIU 90 which supplies instruction data from external memory")*
- indicating a top-most valid instruction data unit, a bottom-most valid instruction data unit, and a current instruction data unit to execute with pointers; *(Column 11, lines 17-23 – State the following: "In this*

example, program flow occurs top to bottom with arbitrary change of flow (COF) instructions occurring in address space 7e00f-7e010 and in address space 7e040-7e047. To simplify explanation, the address space along the cache line in instruction cache 65 is broken down into eight-byte wide "quadrants" designated A, B, C, and D". Column 2, lines 38-41 – State the following: "The index tag signals replacement logic in the instruction cache not to replace any cache line that holds instruction data which is currently active in the execution pipeline")

- reading a number of instruction data units from the L0 storage structure, starting at the memory location of the current instruction data unit pointer; *(Column 5, lines 26-30 – State the following: "In ID1, the length of two instructions is decoded (one each for the X and Y execution pipelines) to obtain the X and Y instruction pointers--a corresponding X and Y bytes-used signal is sent back to the prefetch buffer 30 (which then increments for the next sixteen byte transfer)")*
- and rearranging the read instruction data units to align the instruction data units in preparation for execution of an instruction. *(Column 4, lines 44-46 – State the following: "The FPU 70 includes a load/store stage with four-deep load and store queues, a conversion stage (thirty-two bit to eighty bit extended format), and an execution stage")*

Claim 22. A method according to claim 21, wherein the instruction data unit comprises a number of bits equal to the number of bits in a smallest instruction size supported in a processor that supports variable instruction sizes. *(Column 2, lines 60-63 – State the following: "Another feature of the present invention is accommodation of variable length instruction sets with the use of logical OR circuitry to OR multiple (cache line origin) index tags which describe a single instruction.")*

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claim 7 is rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as applied to claims 1 and 6 above, further in view of Ellis et al. (US Patent 5,418,973).

Quattromani teaches the limitations of claims 1 and 6 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to pointers being set equal.

Quattromani fails to teach claim 7, which states "An instruction memory according to claim 6, wherein a read request outside the window of valid instruction data causes the memory control logic to set the pair of pointers equal to each other to invalidate the data in the memory locations." However, Ellis' invention discloses "The queue, for example, is initially empty when the head an tail pointers are equal" (Column 28, lines 9-11). It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Ellis'"Digital Computer System with Cache Controller" before him at the time the invention was made, to combine the inventions to allow the pointers to be equal to denote invalidation so that the system would run more accurately and efficiently.

5. Claim 8 is rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as and Ellis as applied to claims 1 and 6-7 above, further in view of Gupta et al. (US Patent 5,941,983 A1).

Quattromani teaches the limitations of claims 1 and 6-7 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to forward jumping.

Quattromani fails to teach claim 8, which states "An instruction memory according to claim 7, further comprising determining if the read request outside the window of valid instruction data jump is a forward jump to an instruction address whose address offset from the current instruction address is less than a threshold, and if the address offset is less than the threshold, continuing to fetch instructions until the instruction of the forward jump has been fetched." However, Gupta's invention discloses "If the branch condition is satisfied, the functional unit sets the instruction pointer for the queue to the program counter value encoded in the branch instruction. This program counter value gives the instruction fetch unit the target instruction within the same queue as the branch instruction. The branch displacement can be positive or negative, meaning that the branch can jump forward or backward in the queue. The processor handles branch instructions in each queue in a similar manner. Each of the branch instructions that the original branch instruction was split into specify the location to jump to within the respective queues." (Column 20, lines 60-67 and Column 21, lines 1-3). It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Gupta's "Out-of-order execution using encoded dependencies between instructions in queues to determine stall values that control issuance of instructions from the queues" before him at the time the invention was made, to allow for forward jumping so that the system would run more quickly and efficiently.

6. Claims 12-13 are rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as applied to claims 9 and 11 above, further in view of Cai et al. (US Patent 5,835,949).

Quattromani teaches the limitations of claims 9 and 11 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to loops or loop buffers.

Quattromani fails to teach claims 12-13, which respectively state "A memory structure according to claim 11, wherein the address indicators comprise an address indicator to indicate the top instruction of a loop and an address indicator to indicate the bottom instruction of a loop", and A memory structure according to claim 9, wherein the memory locations to store instruction data comprises the memory locations to store at least the first instructions of a software loop, and the control logic to provide control signals comprises the control logic to provide control signals to cause the reading and execution of the stored first instructions of the software loop as a zero-overhead loop buffer." However, Cai's invention discloses that "The present embodiments relate to microprocessors, and are more particularly directed to microprocessor circuits, systems, and methods implementing a loop and/or stride predicting load target buffer." (Column 1, lines 12-15). Cai further states "Continuing with the above example from Table 1, instruction 14 returns control to instruction 10 for the next iteration where J=2. For a second time, therefore, instruction 11 is fetched by instruction fetch stage 40. Again it is detected that instruction 11 is a data fetching instruction and, therefore, LTB 56 is consulted to determine whether one of its entries corresponds to instruction 11. In the current example, because of the previous incident of instruction 11, and assuming no other intervening event has affected the entry, then the ADDRESS TAG of entry 56.sub.1 is detected as corresponding to instruction 11. In response, the NEXT POINTER of entry 56.sub.1 is consulted to determine what the predicted target data address is for instruction 11. Recall that the NEXT POINTER

indicates POINTER B; thus, the value of POINTER B is examined as a potential target data address. Recall further that the B CONTROL corresponding to POINTER B is currently set to invalid. Thus, at this point, there is no prediction of a target data address for the second incident of instruction 11. In response, first the value of the NEXT POINTER is set to 10, that is, to indicate that POINTER C is the next pointer to be consulted upon the next incident of instruction 11. Moreover, again as a default, it is predicted that the current data fetching instruction (i.e., instruction 11) will be part of a loop mode. Thus, since it is assumed that a loop will be formed from the data target address of POINTER B toward the data target address of POINTER C, the B CONTROL is set to 110, thereby predicting a loop mode where the next target address is pointed to by POINTER C. Thereafter, again the process waits for instruction 11 to pass through pipeline 38 until its actual target data address is determined. In the example shown in connection with FIG. 3, recall that the second incident of instruction 11 requires data from address 2214; thus, the address of 2214 is stored into POINTER B. Note after two target data addresses are stored in two POINTERs (e.g., POINTER A and B) as in the example thus far (or as an alternative either at the same time the second address is being stored or before it is being stored), an additional test is performed for reasons more clear below. Specifically, it is determined whether the two target data addresses match. In the current example, however, there is no such match. Thus, the process continues under the assumption of the loop mode as detailed below" (Column 18, lines 1-41).

It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Cai's"Microprocessor Circuits, Systems, and Methods of Implementing a Loop and/or Stride Predicting Load Target

Buffer" before him at the time the invention was made, to combine the inventions to include loops and loop buffers so that the system would be more diversified, accurate and user-friendly.

7. Claim 23 is rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as applied to claim 21 above, further in view of Munson (US Patent 6,408,377 B2).

Quattromani teaches the limitations of claim 21 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to reading a number of instruction data units equal to a maximum number of instruction data units in an instruction latch.

Quattromani fails to teach claim 23, which states "A method according to claim 21, wherein reading the number of instruction data units comprises reading a number of instruction data units equal to a maximum number of instruction data units that an instruction latch in a processor instruction sequencer can hold simultaneously." However, Munson states "These instructions are then arranged by the instruction issue (IS) stage in their order of execution. Shown in the IS stage of FIG. 4 are six latches 217-222, each of which is capable of storing the maximum number of bytes forming any instruction that is expected to be received by the stage" (Column 9, lines 61-65). It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Munson's "Dynamic allocation of resources in multiple microprocessor pipelines" before him at the time the invention was made, to combine the inventions to allow reading a number of instruction data units equal to a maximum number of instruction data units that an instruction latch so that the system would run more efficiently.

8. Claim 24 is rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as applied to claims 21 and 23 above, further in view of Ross et al. (US PG Pub 2005/0147203 A1).

Quattromani teaches the limitations of claims 21 and 23 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to reading the number of instruction data units starting from a memory location in the center of a row of memory locations.

Quattromani fails to teach claim 24, which states "A method according to claim 23, wherein reading from the memory location of the current instruction data unit comprises reading the number of instruction data units starting from a memory location in the center of a row of memory locations, and reading the instruction data units stored in all memory locations in the row from the location in the center to the last memory location in the row, and reading at least one instruction data unit from a memory location in the next row." However, Ross states "The article of claim 28, wherein said instructions, when executed, further result in said data in said determined area of interest being read by row, starting from the center of said panel and reading towards rows remote from said center" (Claim 30, lines 1-5). It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Ross' "Reading Imaging Data" before him at the time the invention was made, to combine the inventions to allow reading the number of instruction data units starting from a memory location in the center of a row of memory locations so that the system would run more accurately and efficiently.

9. Claim 25 is rejected under 35 U.S.C.103(a) as being unpatentable over Quattromani as applied to claims 21, and 23-24 above, further in view of Hershey et al. (US Patent US 5,239,584 A)

Quattromani teaches the limitations of claims 21, and 23-24 for the reasons above.

Quattromani's invention differs from the claimed invention in that there is no specific reference to specifically shifting the read instruction data units.

Quattromani fails to teach claim 25, which states "A method according to claim 24, wherein rearranging comprises shifting the read instruction data units to order the instruction data units with the current instruction data unit to occupy the least significant bytes of the sequencer." However, Hershey states "In time broadcast commands, the initialization vector consists of a four byte sequence number. The two least significant bytes of the sequence number number are sent as two bytes of the authentication field of the APDU header. The two most significant bytes are contained in the data field of the metering time broadcast command. These bytes are sent least significant byte first as shown in FIG. 3B" (Column 5, lines 14-21). It would have been obvious to one of ordinary skill in the art, having the teachings of the "Method of Identifying and Self-Modifying Code" of Quattromani and Hershey's "Method and apparatus for encryption/authentication of data in energy metering applications" before him at the time the invention was made, to combine the inventions to allow shifting the read instruction data units to order the instruction data units with the current instruction data unit to occupy the least significant bytes of the sequencer so that the system would run more efficiently.

Conclusion

10. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lev I. Iwashko whose telephone number is (571)272-1658. The examiner can normally be reached on M-F (alternating Fridays), from 8-4PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Matt Kim can be reached on (571)272-4182. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Lev Iwashko

PRIMARY EXAMINER
GROUP 2100